

Programmation logique

Damien Nouvel



Plan

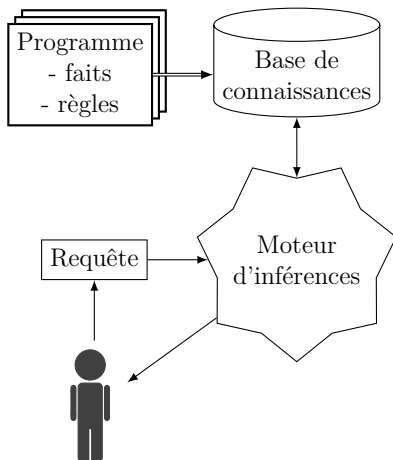
1. Principes de programmation logique
2. Programmation logique avec SWI-Prolog
3. Mécanismes de Prolog

Historique de la programmation logique

- ▶ Historiquement : calculateurs, programmation **impérative**
 - Fonctionnement pas-à-pas
 - Compilation mais pas d'**inférences**
- ▶ Manipulation de **connaissances**
 - Déclarations de variables et fonctions
 - Interrogation par requêtes (non-déterministe)
 - Règles d'inférences

⇒ Moins à implémenter, mais plus contraignant
- ⇒ **Prolog** : PROgrammation LOGique (Colmerauer, 1972)
 - **Chaînage arrière** à partir de buts
 - **Unification** par substitutions

Fonctionnement général



Syntaxe Prolog

- ▶ Briques élémentaires
 - **Constantes** avec une minuscule : pierre, paul
 - **Nombres** entier ou flottant : 42, 2.3
 - **Chaînes** entre guillemets : "abcdef"
 - **Variables** avec une majuscule : X, Y
 - **Prédicats** avec une minuscule : humain(X)
- ▶ Assertions ou “instructions” (terminent par un point)
 - Faits / base de connaissances, par exemple
 - Prédicat unaire : humain(pierre).
 - Prédicat binaire : parent(pierre, paul).
 - Règles : fils(X, Y) :- parent(Y,X), homme(X).
 - Requêtes : ?-enfant(paul, X).
 - Sans variables, réponse dans {V, F} (true, fail)
 - Avec n variables, réponse dans {F} \cup D^n
 - Différent de : X \= Y

Plan

1. Principes de programmation logique
2. Programmation logique avec SWI-Prolog
3. Mécanismes de Prolog

Généralités

- ▶ SWI-Prolog
 - Jan Wielemaker (1987)
 - Environnement en ligne de commande
 - Demo en ligne : <http://swish.swi-prolog.org/>

Exercice

▸ Famille

- Modélisez les faits suivant
 - Jean et Marie sont parents de Pierre, Paul et Sophie
 - Pierre a deux enfants : Eliott et Rosie
 - Sophie a trois enfants : Adrien, Irène et Marc
- Définissez des règles pour modéliser les relations
 - Grand-parent
 - Petit-enfant
- Ajoutez les faits sur le sexe de chaque personne et ajoutez
 - Frère, sœur
 - Oncle, tante
- Faites les requêtes qui retournent
 - Les oncles de Marc
 - Les frères d'Irène
 - Le grand-père de Rosie

Plan

1. Principes de programmation logique
2. Programmation logique avec SWI-Prolog
3. Mécanismes de Prolog

Unification

- ▶ Comment trouver une réponse à une requête ?

⇒ Unifier des clauses avec la requête

- **Substitutions** de **variables** adéquates dans les **faits**
- Exemple
 - `pere(jean, pierre).`
 - `?- pere(jean, X).`
 - $pere(Jean, X)[X/pierre]$

⇒ Faire des inférences

- **Substitutions** de **variables** adéquates dans les **règles**
- Exemple
 - `pere(pierre, eliott).`
 - `gdpere(X,Y) :- pere(X, Z), pere(Z, Y).`
 - `?- gdpere(jean, X).`
 - $pere(X, Z) \wedge pere(Z, Y)[X/jean, Y/eliott, Z/pierre]$
 - $\vdash gdpere(Jean, Eliott)$
 - $\equiv gdpere(Jean, X)[X/Eliott]$

Opérateur de négation, d'unification, de listes

- ▶ L'opérateur `not` pour la négation
 - `impair(X) :- entier(X), not(pair(X))`
- ▶ L'opérateur `is` force l'unification entre expressions
 - `X is 3*2`
- ▶ Les **listes** pour les ensembles ordonnées
 - Liste vide : `[]`
 - Liste de trois éléments : `[a, b, c]`
 - Tête et queue de la liste : `[Tete|Queue]`
 - `enfants(X, [Aine|AutresEnfants])`

Récurtivité

- ▶ Forte utilisation de la récursivité
 - Sous forme logique, deux clauses
 - Famille : fonction ancêtre (parent du parent du parent ...)
 - `ancetre(X,Y) :- parent(X,Y).`
 - `ancetre(X,Y) :- parent(X,Z), ancetre(Z,Y).`
- ⇒ Clause initiale et récursivité sur un prédicat
- ⇒ Attention à l'ordre des clauses

Exercice

▸ Animaux

- Définissez un ensemble d'être vivants comme
 - Animaux
 - Végétaux
- Définissez
 - Un prédicat `mange(X,Y)` . sur ces individus
 - Le prédicat `predateur(X,Y)`
 - Le prédicat `carnivore(X)`
 - Le prédicat `herbivore(X)`
 - Le prédicat `omnivore(X)`

Exercice

► Cuisine

- Le restaurant des plats (avec leurs ingrédients)
 - Salade (laitue, oignons, pain, huile)
 - Tomates (tomates, huile)
 - Pâté (canard, pain)
 - Lasagne (pâte, boeuf, oignons, tomates)
 - Légumes (patates, carottes, oignons, pain)
 - Escalope (patates, poulet)
 - Salade de fruits (pomme, banane, raisin)
 - Tarte (pâte, pomme)
 - Fondant (chocolat)
- Définissez
 - Les entrées, les plats et les desserts
 - Un menu rapide : entrée / plat ou plat / dessert
 - Un menu complet : entrée, plat, dessert
 - Les plats qui contiennent de la viande
 - Les menus qui contiennent de la viande

Test prolog

```
somePredicate(A, B) :-  
  arbitraryPredicate(A, _, 1, 2),  
  predicateWithAtom(someAtom),  
  anotherPredicate(B, someAtom, myPredicate(A, _)),  
  findall(X, ('testString'(X), myPredicate(A, X)), L1),  
  member(A, L1),  
  !.
```